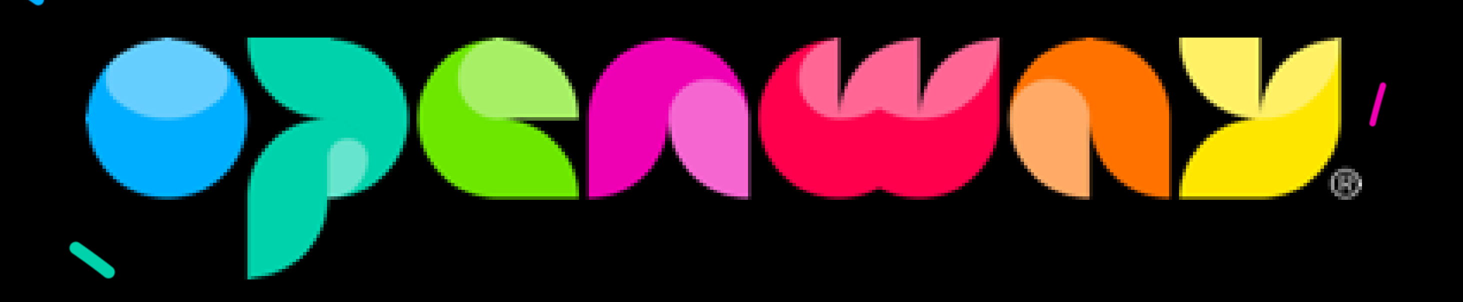
Кошелек с нуля в 2020 году: технологии, вызовы, решения

Банальности, проверенные опытом

Дельгядо Филипп







Международный поставщик ПО для банков и финтехов

Лидер рынка по оценке Aite, Gartner, Ovum

Решения работают в 77 странах мира

15% карт и магазинов в Европе

88% банковских карт в России

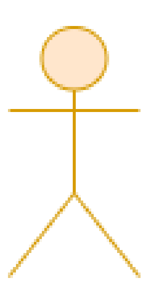
ЭМИССИЯ

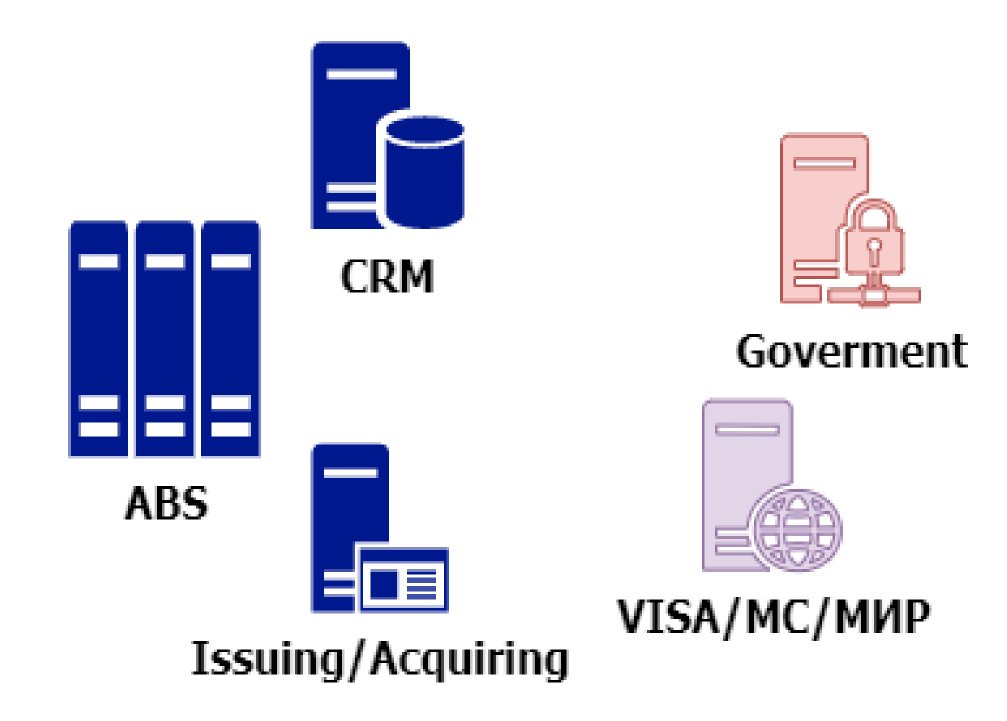
ЭКВАЙРИНГ

МОБИЛЬНЫЕ КОШЕЛЬКИ

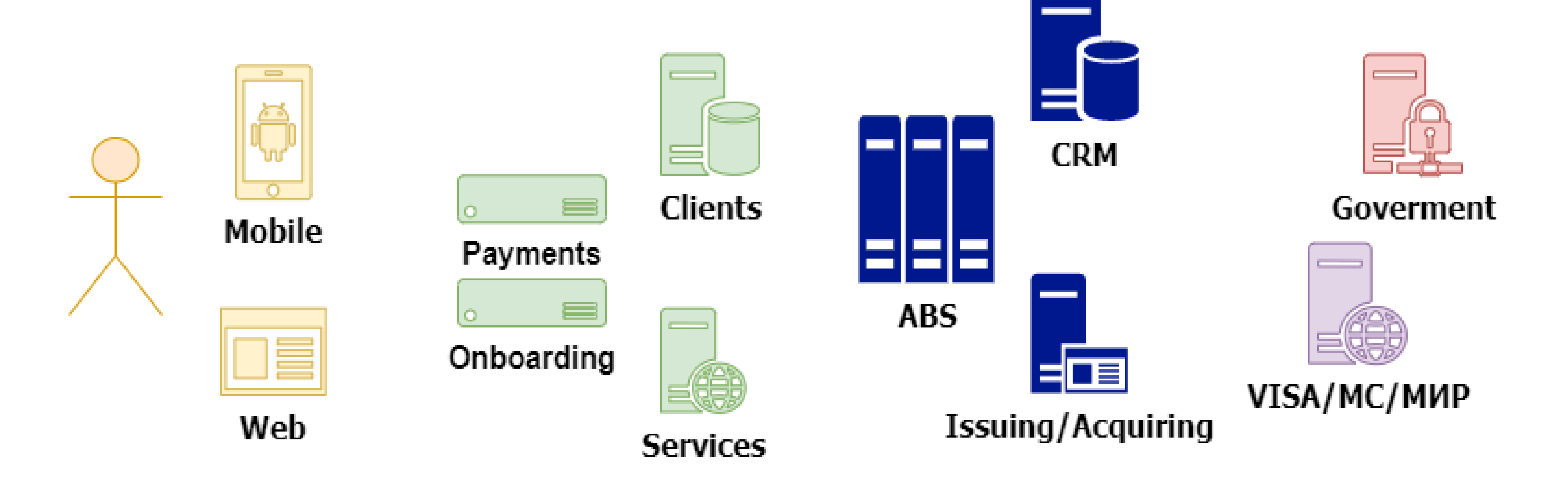
ПРОЦЕССИНГ ТРАНЗАКЦИЙ

Что такое кошелек





Что такое кошелек



Кому нужны кошельки

Банкам

Национальным платежным системам (СБП и похожим)

Крупным торговым сетям

Закрытым системам расчетов (PayPal, ЮMoney, …)

Какие клиенты бывают

Большие - 100+ млн клиентов, 5 000 платежей в секунду
Маленькие - 1 млн. клиентов, 50 платежей в секунду

Какие клиенты бывают

Большие - 100+ млн клиентов, 5 000 платежей в секунду
Маленькие - 1 млн. клиентов, 50 платежей в секунду

Богатые - готовы купить любой софт с поддержкой
Бедные - нет денег даже на нормального DBA

Какие клиенты бывают

Большие	- 100+ млн клиентов, 5 000 платежей в секунду
Маленькие	- 1 млн. клиентов, 50 платежей в секунду
Богатые	- готовы купить любой софт с поддержкой
Бедные	- нет денег даже на нормального DBA
В Европе	- GDPR, много регуляторов
В третьем мире	- законы пишутся на ходу

Какие клиенты бывают

Большие	- 100+ млн клиентов, 5 000 платежей в секунду
Маленькие	- 1 млн. клиентов, 50 платежей в секунду
Богатые	- готовы купить любой софт с поддержкой
Бедные	- нет денег даже на нормального DBA
В Европе	- GDPR, много регуляторов
В третьем мире	- законы пишутся на ходу
Старые	- своя сложившаяся инфраструктура
Новые	- даже не думали про инфраструктуру

Проект

Нужно дешевое, масштабируемое, неприхотливое и кастомизируемое решение.

Проект

Нужно дешевое, масштабируемое, неприхотливое и кастомизируемое решение.

Внутренний стартап:

- свобода в выборе технологий
- свобода в выборе процессов
- 2 pizza team
- доступ к специалистам в компании

Архитектура

Архитектура

**Architecture is about the important stuff.
Whatever that is.**

Ralph Johnson



Архитектура

Архитектура как необходимые выборы

Архитектура как язык

Архитектура как управление сложностью

Архитектура как точки зрения

Выбор БД

OLTP

OLAP

Выбор БД

OLTP:

надежность
масштабируемость
бесплатность
простота эксплуатации
поддержка

Выбор БД

OLTP:

- надежность
- масштабируемость
- бесплатность
- простота эксплуатации
- поддержка

~~Oracle~~
~~Postgress DB~~
~~Mongo~~

Выбор БД

OLTP:

- надежность
- масштабируемость
- бесплатность
- простота эксплуатации
- поддержка

~~Oracle~~
~~Postgress DB~~
~~Mongo~~
Foundation DB

FoundationDB

Open-source distributed scalable transactional (ACID) key-value database with crazy thorough testing

FoundationDB

Open-source distributed scalable transactional (ACID) key-value database with crazy thorough testing

«haven't tested foundation in part because their testing appears to be waaaay more rigorous than mine.»

Kyle Kingsbury (@aphyr)



FoundationDB – особенности

Нет платной поддержки

Низкоуровневый сложный API

Нет ограничений доступа

Слабый инструментарий

FoundationDB – особенности

Нет платной поддержки

предоставляем клиентам поддержку самостоятельно

Низкоуровневый сложный API

Нет ограничений доступа

Слабый инструментарий

FoundationDB – особенности

Нет платной поддержки

предоставляем клиентам поддержку самостоятельно

Низкоуровневый сложный API

написали свою библиотеку с поддержкой асинхронного доступа, типизацией, сериализацией в CBOR, поддержкой house-keeping

Нет ограничений доступа

Слабый инструментарий

FoundationDB – особенности

Нет платной поддержки

предоставляем клиентам поддержку самостоятельно

Низкоуровневый сложный API

написали свою библиотеку с поддержкой асинхронного доступа, типизацией, сериализацией в CBOR, поддержкой house-keeping

Нет ограничений доступа

добавили в библиотеку криптографию и управление ключами

Слабый инструментарий

FoundationDB – особенности

Нет платной поддержки

предоставляем клиентам поддержку самостоятельно

Низкоуровневый сложный API

написали свою библиотеку с поддержкой асинхронного доступа, типизацией, сериализацией в CBOR, поддержкой house-keeping

Нет ограничений доступа

добавили в библиотеку криптографию и управление ключами

Слабый инструментарий

сделали свой движок миграций типа Flyway

Выбор БД

OLTP:

FoundationDB

OLAP:

Выбор БД

OLTP:

FoundationDB

OLAP:

Clickhouse

Архитектура

Архитектура как необходимые выборы

Архитектура как язык

Архитектура как управление сложностью

Архитектура как точки зрения

Прочие выборы

Polyglot development vs one lang to rule them all

Kotlin vs C# vs Java

Flutter vs React Native

REST 1 vs REST 2 vs gRPC

…

Прочие выборы

Polyglot development vs **one lang to rule them all**

Kotlin vs C# vs Java

Flutter vs React Native

REST 1 vs REST 2 vs gRPC

…

Архитектура

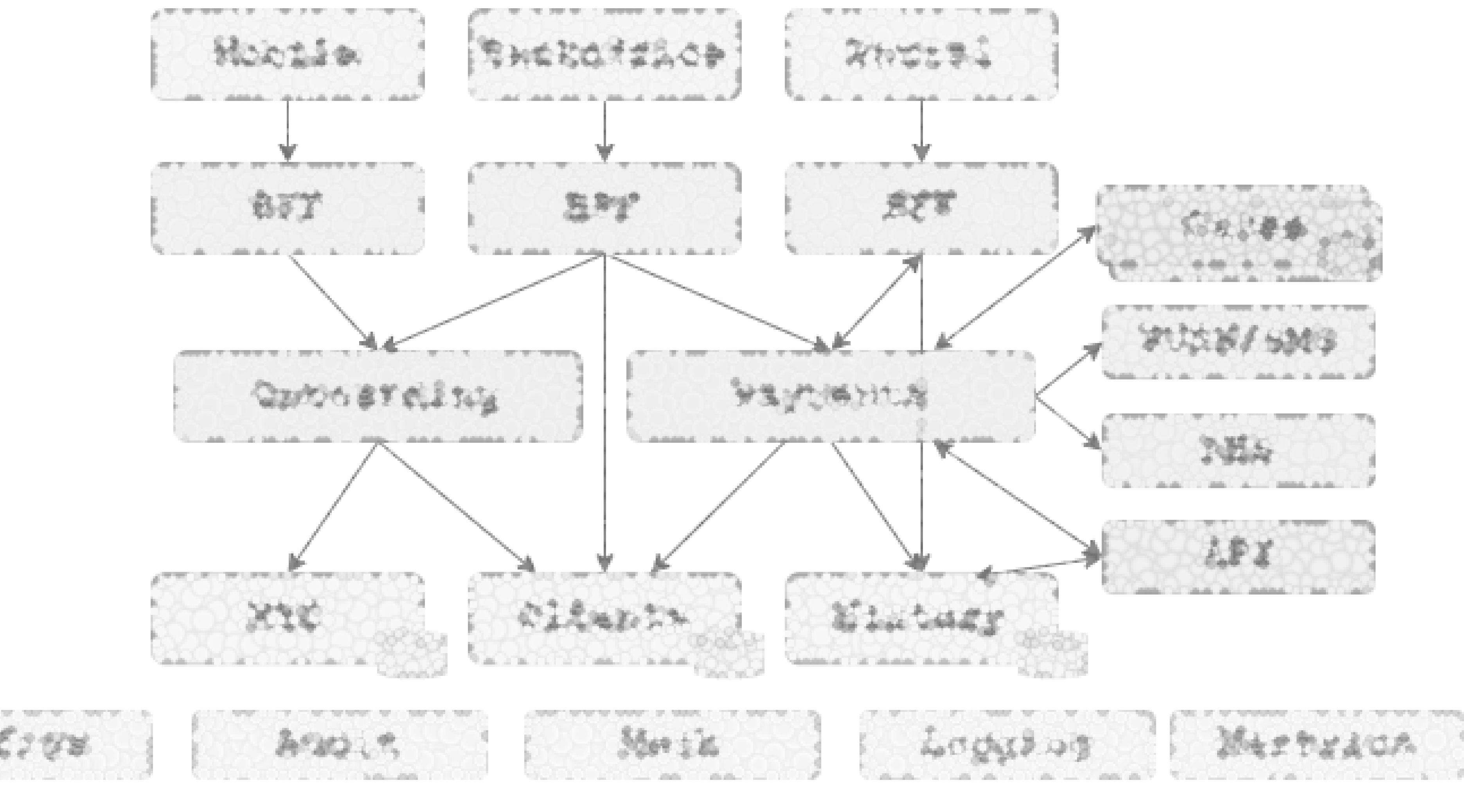
Архитектура как необходимые выборы

Архитектура как язык

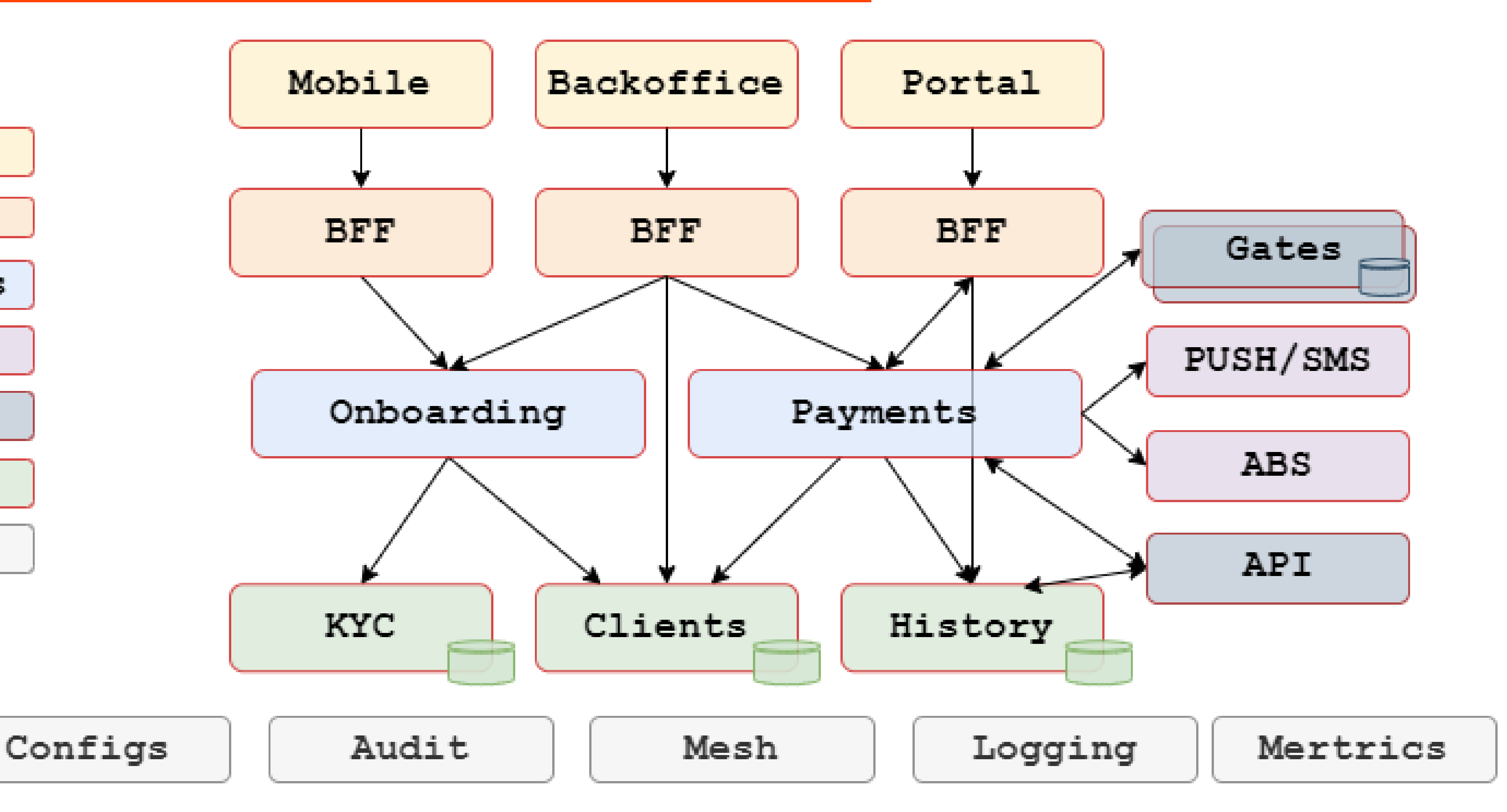
Архитектура как управление сложностью

Архитектура как точки зрения

Архитектура сервисов



Архитектура сервисов



Архитектура

Архитектура как необходимые выборы

Архитектура как язык

Архитектура как управление сложностью

Архитектура как точки зрения

Бизнес-сценарии

Онбординг:

3-10 шагов, 3-5 подсистем, десятки сервисов

Платеж:

2-5 шагов, 2-5 подсистем, несколько сервисов

Блокировки, KYC management, etc.

Саги

Разбиваем бизнес-процесс на множество локальных шагов

Для каждого шага проектируем компенсацию

Фиксируем список выполненных шагов

В случае ошибки запускаем все компенсации на пройденные шаги

https://microservices.io/patterns/data/saga.html
https://www.cs.cornell.edu/andru/cs711/2002fa/reading/sagas.pdf

Саги?

Обычно нужно не отменить процесс, а дожать его другим способом



Саги?

Обычно нужно не отменить процесс, а дожать его другим способом

Компенсации не всегда возможны

Саги?

Обычно нужно не отменить процесс, а дожать его другим способом

Компенсации не всегда возможны

Компенсация нужна не для операции, а для всего процесса

Саги?

Обычно нужно не отменить процесс, а дожать его другим способом

Компенсации не всегда возможны

Компенсация нужна не для операции, а для всего процесса

Сложно учитывать ограничения всей бизнес-транзакции (время выполнения, бюджет транзакции и т.п.)

Саги?

Обычно нужно не отменить процесс, а дожать его другим способом

Компенсации не всегда возможны

Компенсация нужна не для операции, а для всего процесса

Сложно учитывать ограничения всей бизнес-транзакции (время выполнения, бюджет транзакции и т.п.)

Код получается довольно сложный

Workflow

Описываем процесс как императивный код из отдельных шагов

При выполнении сохраняем результат каждого шага

При проблемах с шагом – стараемся его повторять

При падении сервера повторяем весь процесс

Для исключительных ситуаций пишем такой же workflow



Haydn4k

Библиотека, а не сервис

С **удобным** DSL

С поддержкой более 100 000 одновременных сценариев

С поддержкой до 20 000 шагов в секунду

Foundation DB + Kotlin

Haydn4k

```
suspend fun onboard(data: OnboardingData) : ClientId = play {
  val identityId = step { identityService.nextId ()  }
  step { identityService.create (identityId, data.credentials)  }
  val clientId = step { clientService.create (identityId, data.clientInfo) }
 return@play clientId
}.onError {
  step { if (clientId!=null) clientService.remove(clientId) }
  step { if (identityId!=null) identityService.remove(identityId) }
}.async()
```

Идемпотентность и повторяемость

“Идемпотентность – свойство операции при повторном применении давать тот же результат, что и при первом”

Идемпотентность и повторяемость

“Идемпотентность – свойство операции при повторном применении давать тот же результат, что и при первом”

Повторяемость – возможность повторять операцию без нежелательных последствий

Haydn4k

```
suspend fun onboard(data: OnboardingData) : ClientId = play
(uniqueId = {data.phone}) {
  deadline = now().plusSeconds(30)
  val identityId = step { identityService.nextId ()  }
  step { identityService.create (identityId, data.credentials)  }
  step( policy = {rpsLimit = 10 })
          { govService.checkCitizen(data.fio, data.document, uniqueId)  }
  val isCitizen = waitSignal { it is CheckCitizenComplete }
  If (!isCitizen) step { fbiService.report (data.fio, data.document)  }
  val clientId = step { clientService.create (identityId, data.clientInfo) }
 return@play clientId
}.await()
```

Haydn4k – возможности

Длительная заморозка

Все возможности Resilience4J
circuit-breaker, limits etc

Мягкое восстановление при сбоях

Сложная обработка ошибок

Интеграция с ktor

Метрики и логирование

Haydn4k – производительность

Примерно 2000 шагов в секунду на экземпляр FDB

1 core + 4GB + 4000 IOPS

Примерно горизонтальное масштабирование

Можно подменить FDB на что-то побыстрее

Queue

Сигналы для Haydn4k

События, отправляемые с сервера на клиент

- Server Sent Messages

Внутренние события

- Отправка SMS, отчеты по расписанию и т.п.

Очереди пользовательских задач

- Изменения параметров кошелька и т.п.

Queue – требования

10+ млн. очередей

0 - 1000 событий в очереди

50K+ событий в секунду

Гарантии at least once

10 mln topic problem

Kafka
- не больше 200k partitions

РСУДБ
- не больше 10K событий в секунду

RabbitMQ
- не больше сотен тысяч очередей

Queue – решение

FoundationDB – хранение самих очередей

- в одной записи несколько событий
- несколько (обычно одна) записей на очередь
- house-keeping

Kafka – нотификации по изменениям

- воркеры узнают, какая очередь изменилась
- читают целиком из FDB, сохраняют изменения
- очереди распределяются по партициям

Архитектура

Архитектура как необходимые выборы

Архитектура как язык

Архитектура как управление сложностью

Архитектура как точки зрения

Архитектура продукта

Стратегии конфигурации и кастомизации

Параметры конфигурации

Точки изменения функциональности

Точки расширения функциональности

Кастомизации

Стратегии конфигурации и кастомизации

Параметры конфигурации

Точки изменения функциональности

Точки расширения функциональности

Кастомизации

Kotlin DSL

Kotlin DSLs

Код на kotlin

DSL для конкретной задачи

Хранится в общем config service

Компилируется при старте

Выполняется при необходимости

Включает инструменты для написания скриптов

- настроенная ide, скрипты сборки, скрипты для тестов и т.п.

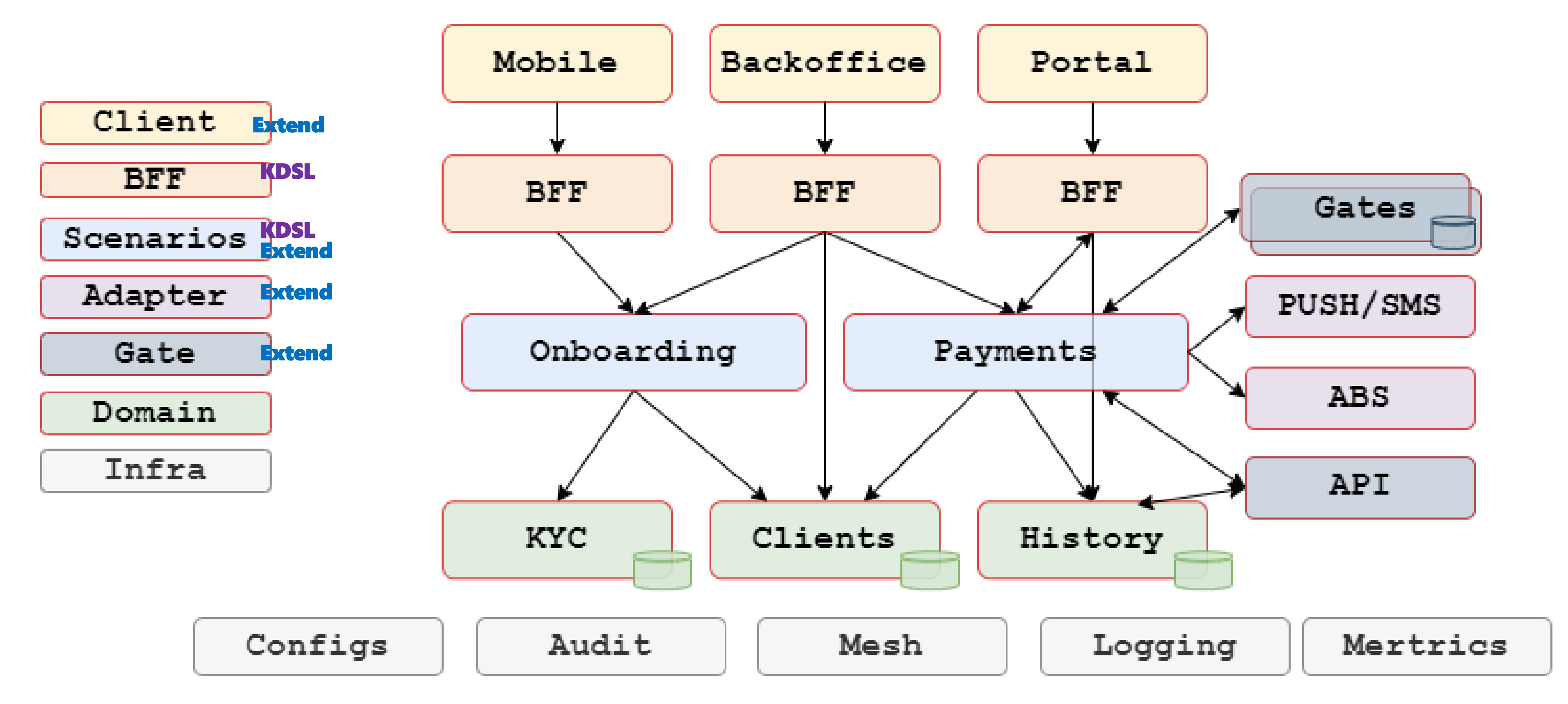
Kotlin DSLs

Формирование описания платежа

Специфические сценарии
блокировки, изменение документов, онбординг

Проверка правил доступа (ABAC)

Архитектура продукта



Другие viewpoints

Кастомизация

Авторизация/Аутентификация

Связность сервисов

Безопасность

Высокая доступность

Интернационализация

Обеспечение качества

Авторизация / Аутентификация

10+ млн внешних пользователей

10 000 внутренних пользователей

Разные механизмы подтверждения операций

Разные модели доступа

Авторизация / Аутентификация

Самописный сервис IAM/CIAM

с возможностью подключать внешние IAM

Множество разных сценариев аутентификации

RBAC через конфигурацию

ABAC через Kotlin DSL

Связность сервисов – требования

Работа в чужой инфраструктуре

Простота в развертывании и в поддержке

Реализация HA/LB

Централизованное управление конфигурацией

Связность сервисов

Собственный service-mesh

service-locator, load balancing, heartbeat, lua scripts etc

Написан на Java+Netty

13 000 rps на ядро

Средняя задержка 0.4 ms

Итого

Kotlin + Foundation DB

Kotlin API first development

Собственный workflow engine, service mesh

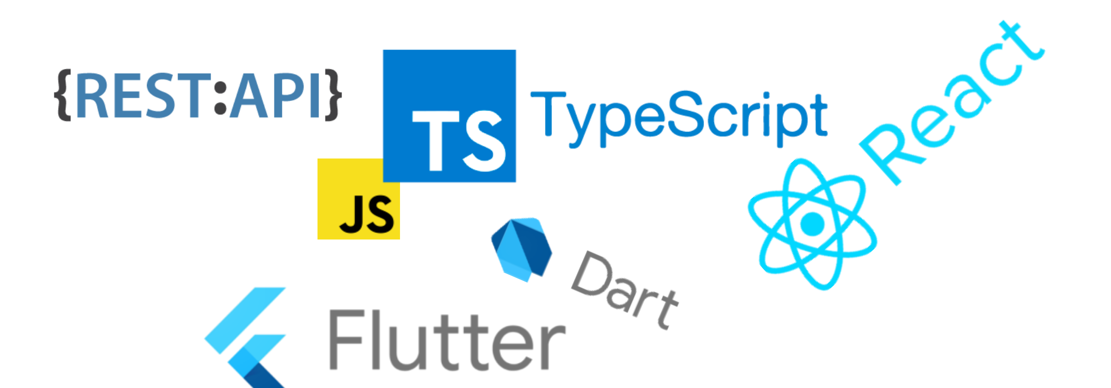
Kotlin DSLs

Libraries over services

Итого







Kanban





Итого

Архитектура как необходимые выборы

Архитектура как язык

Архитектура как управление сложностью

Архитектура как точки зрения

Итого

Архитектура как необходимые выборы

Архитектура как язык

Архитектура как управление сложностью

Архитектура как точки зрения

Архитектура как непрерывный процесс

Архитектура как организация

Архитектура как ограничения

Архитектура как гарантии

Вопросы?

Дельгядо Филипп

dph.main@gmail.com
vk.com/dphil
telegram: @dphil

